What is claimed is:

1. A method comprising:

storing a firmware module in memory, wherein the firmware module follows a portable executable (PE) format having subdivisions that include an MS-DOS header; and

flattening the firmware module by replacing existing content within at least one field within the MS-DOS header of the firmware module with fill data that is more compressible than the existing content.

2. A method according to claim 1, wherein the operation of flattening the firmware module comprises loading fill data into at least fifty bytes of the MS-DOS header.

3. A method according to claim 1, wherein the operation of flattening the firmware module comprises loading fill data into an MS-DOS stub field within the MS-DOS header.

4. A method according to claim 1, wherein the operation of flattening the firmware module comprises ensuring that fill data occupies all fields within the MS-DOS header except for an lfanew field and an e-magic field.

5. A method according to claim 1, wherein the PE format also includes an optional file header, the method further comprising:

loading fill data into at least one field within the optional file header.

6. A method according to claim 5, wherein the operation of loading fill data into at least one field within the optional file header comprises:

loading fill data into at least one of a SizeOfStackReserve field, a SizeOfStackCommit field, a SizeOfHeapReserve field, a SizeOfHeapCommit field, and a LoaderFlags field.

7. A method according to claim 1, further comprising:

merging at least two sections from an object file into one section in the firmware module.

8. A method according to claim 7, wherein the operation of merging at least two sections from an object file into one section in the firmware module comprises

instructing a linker to merge the at least two sections when generating the firmware module from the object file.

9. A method according to claim 8, further comprising:

causing the linker to change a name of a section specified in the object file to a more compressible name.

10. A method according to claim 1, wherein the PE format also includes an image page, the method further comprising:

storing, in the image page, an alternate file path for a debug file associated with the firmware module, wherein the alternate file path is more compressible than an original file path for the debug file.

11. A method according to claim 1, wherein the PE format also includes an image page, the method further comprising:

instructing a linker to store, in the image page of the firmware module, an alternate file path for a debug file associated with the firmware module, wherein the alternate file path is more compressible than an original file path for the debug file.

12. A program product comprising:

a machine accessible medium; and

instructions encoded in the machine accessible medium, wherein the instructions, when executed by a processing system, cause the processing system to perform operations comprising:

accessing a firmware module within the processing system, wherein the firmware module follows a portable executable (PE) format having subdivisions that include an MS-DOS header; and

flattening the firmware module by replacing existing content within at least one field within the MS-DOS header of the firmware module with fill data that is more compressible than the existing content.

13. A program product according to claim 12, wherein the operation of flattening the firmware module comprises loading fill data into at least fifty bytes of the MS-DOS header.

14. A program product according to claim 12, wherein the operation of flattening the firmware module comprises loading fill data into an MS-DOS stub field within the MS-DOS header.

15. A program product according to claim 12, wherein the operation of flattening the firmware module comprises ensuring that fill data occupies all fields within the MS-DOS header except for an lfanew field and an e-magic field.

16. A program product according to claim 12, wherein the PE format also includes an optional file header, the program product further comprising:

instructions which, when executed by the processing system, cause the processing system to load fill data into at least one field within the optional file header.

17. A program product according to claim 16, wherein the operation of loading fill data into at least one field within the optional file header comprises:

loading fill data into at least one of a SizeOfStackReserve field, a SizeOfStackCommit field, a SizeOfHeapReserve field, a SizeOfHeapCommit field, and a LoaderFlags field.

18. A processing system with resources for flattening a firmware module, the processing system comprising:

a processor;

memory communicatively coupled to the processor; and

instructions stored in the memory, wherein the instructions, when executed by the processor, cause the processing system to perform operations comprising:

accessing a firmware module within the processing system, wherein the firmware module follows a portable executable (PE) format having subdivisions that include an MS-DOS header; and

flattening the firmware module by replacing existing content within at least one field within the MS-DOS header of the firmware module with fill data that is more compressible than the existing content.

19. A processing system according to claim 18, wherein the operation of flattening the firmware module comprises loading fill data into at least fifty bytes of the MS-DOS header.

20. A processing system according to claim 18, wherein the operation of flattening the firmware module comprises loading fill data into an MS-DOS stub field within the MS-DOS header.

21. A processing system according to claim 18, wherein the operation of flattening the firmware module comprises ensuring that fill data occupies all fields within the MS-DOS header except for an lfanew field and an e-magic field.

22. A processing system according to claim 18, wherein the PE format also includes an optional file header, the processing system further comprising:

instructions which, when executed by the processor, cause the processing system to load fill data into at least one field within the optional file header.

23. A processing system according to claim 22, wherein the operation of loading fill data into at least one field within the optional file header comprises:

loading fill data into at least one of a SizeOfStackReserve field, a SizeOfStackCommit field, a SizeOfHeapReserve field, a SizeOfHeapCommit field, and a LoaderFlags field.

24. An apparatus comprising:

a machine accessible medium; and

a firmware module encoded in the machine accessible medium, the firmware module having a portable executable (PE) format with subdivisions that include an MS-DOS header, wherein the firmware module was produced by operations comprising:

flattening the firmware module by replacing existing content within at least one field within the MS-DOS header of the firmware module with fill data that is more compressible than the existing content.

25. An apparatus according to claim 24, further comprising:

a processor communicatively coupled to the machine accessible medium;

memory communicatively coupled to the processor; and

instructions stored in the memory, wherein the instructions, when executed by the processor, cause the processing system to perform operations comprising:

retrieving the firmware module from the machine accessible medium; and

executing the firmware module within a boot environment.

26. An apparatus according to claim 24, wherein:

the machine accessible medium comprises a non-volatile storage device; and

the apparatus further comprises an interface in communication with the non-volatile storage device, the interface operable to provide communication between the non-volatile storage device and a processor of a data processing system.

27. An apparatus according to claim 26, wherein the apparatus comprises an adapter card for a processing system.